Received at NIC 22-APR-74

30489

Network Working Group
Request for Comments #635
NIC #30489

RECEIVED
MAY 0 1 1974

Postel

An Assessment of ARPANET Protocols

Vinton G. Cerf
Stanford University

ABSTRACT

This paper presents some theoretical and practical motivations for the redesign of the ARPANET communication protocols. Issues concerning multipacket messages, Host retransmission, duplicate detection, sequencing, and acknowledgment are discussed. Simplifications to the IMP/IMP protocol are proposed on the assumption that new Host level protocols are adopted. Familiarity with the current protocol designs is probably necessary since many of the arguments refer to details in the present protocol design.

An Assessment of ARPANET Protocols

Vinton G. Cerf

Stanford University

Introduction

The history of the Advanced Research Project Agency resource sharing computer network (ARPANET) [6] is in many ways a history of the study, development, and implementation of protocols. During the early development of the network many important concepts were discovered and introduced into the protocol design effort. Protocol layering (functional separation of different levels of network transmission), the notion of bilateral rendezvous to set up Host-to-Host connections [1,2], and the definition of a Network Virtual Terminal to aid in the specification of a Terminal-to-Host protocol [3,14] are all examples of important early ideas. The tasks facing the ARPANET design teams were often unclear, and frequently required agreements which had never been contemplated before (e.g., common protocols to permit different operating systems and hardware to communicate). The success of the effort, seen in retrospect, is astonishing, and much credit is due to those who were willing to commit themselves to the job of putting the ARPANET together.

Over the intervening five years since the ARPANET was first begun, we have learned a great deal about the design and behavior of the protocols in use. The Imp-to-Imp protocol [4] has undergone continuous revision, and the HOST/IMP interface specification [5] has been modified slightly. In retrospect, and in the light of experience, it seems reasonable to reconsider some of the aspects of the designs and implementations currently in use. Furthermore, the rapid development of national

computer network projects around the world emphasizes the need for international cooperation in the design of communication protocols so that international connections can be accomplished.

This paper deals with the motivations for the redesign of the HOST-to-HOST, IMP-to-IMP, and HOST/IMP communication protocols in the ARPANET. Analyses of theoretical throughput and delay available from existing protocols, and a discussion of some weaknesses in them, are included.

The basic conclusions reached in this report are:

a) Multipacket message facilities can be eliminated without loss of potential throughput, and with a concurrent simplification of IMP software. [8]

b) Ordering by the destination IMP of messages delivered to a destination HOST can lead to a lockup condition similar to the reassembly lockup experienced in an earlier version of the IMP protocol in connection with multipacket message reassembly [7]. Hosts must order arriving messages anyway, so the IMP need not do it.

c) HOST/IMP protocol could be changed to allow arbitrarily long messages to be sent from HOST to IMP, as long as the destination IMP need not reassemble or re-order the arriving packets before delivery to the HOST.

d) Host level retransmission, positive end-to-end acknowledgments, error detection, duplicate detection, and message ordering, can eliminate the need for many of these features in the IMP/IMP protocol, and the Request for next Message (RFNM) facility in the present HOST/IMP protocol.

computer network projects around the world emphasizes the need for international cooperation in the design of communication protocols so that international connections can be accomplished.

This paper deals with the motivation for the redesign of the HOST-to-HOST, IMP-to-IMP, and HOST/IMP communication protocols in the ARPANET. Analyses of theoretical throughput and delay available from existing protocols, and a discussion of some weaknesses in them, are included.

The basic conclusions reached in this report are:

a) Multipacket message facilities can be eliminated without loss of potential throughput, and with a concurrent simplification of IMP software. [8]

b) Ordering by the destination IMP of messages delivered to a destination HOST can lead to a lockup condition similar to the reassembly lockup experienced in an earlier version of the IMP protocol in connection with multipacket message reassembly [7]. HOSTs must order arriving messages anyway, so the IMP need not do it.

c) HOST/IMP protocol could be changed to allow arbitrarily long messages to be sent from HOST to IMP, as long as the destination IMP need not reassemble or re-order the arriving packets before delivery to the HOST.

d) HOST level retransmission, positive end-to-end acknowledgements, error detection, duplicate detection, and message ordering, can eliminate the need for many of these features in the IMP/IMP protocol, and the Request For Next Message (RFNM) facility in the present HOST/IMP protocol.

e) The flow control mechanism in the current HOST-HOST protocol can lose synchronization owing to message loss or duplication.

f) Host level connections should be full duplex.

g) The need for a separate HOST/HOST control connection can be eliminated by carrying control information in the header of each Host transmission.

Throughput Considerations

In spite of the fact that the IMP subnet can deliver up to 80 kb/sec between pairs of Hosts[*], virtually no application using Host software has achieved this figure. An experiment between Tinker and McClellan Air Force Bases in 1971 achieved burst rates as high as 40 kb/sec, but this was achieved by the use of a non-standard Host/Host protocol which transmitted data over multiple logical connections, and which used Host level re-assembly and acknowledgment to achieve reliable, ordered transmission[**]. The following analysis shows that the current Host/Host protocol cannot offer more than 40 kb/sec on a single connection owing to message format overhead, and that this figure drops hyperbolically if the communicating Hosts are separated by several IMPs.

The single major reason for the distance (hop) dependent behavior of Host/Host throughput is the "message-at-a-time" Host/Host protocol. This means that, on a given connection between processes in

---

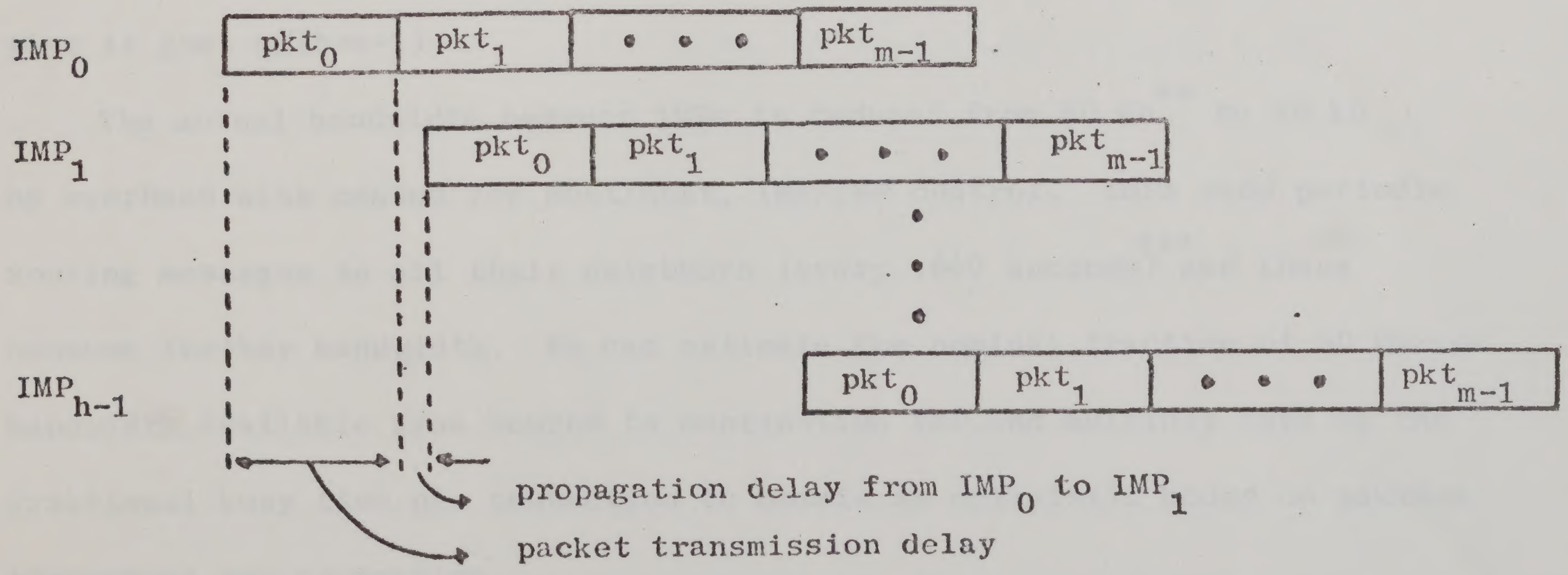
[*] Unpublished measurement experiments at UCLA run by R. Kahn and V. Cerf confirmed this.

[**] Unpublished measurement data obtained by V. Cerf at the ARPA Network Measurement Center, UCLA.

the Hosts, only a single message ranging from 0-8063 bits of data can be outstanding at any moment. When the Host/Host protocol was originally designed, the IMPs provided up to 256 simplex logical links between pairs of Hosts. If a message was sent over a link (there was a one to one relationship between a link and a connection), the link was blocked until a RFNM was received from the destination IMP indicating that the message had been delivered to the Host. Of course, the mechanism was protected by a time-out in case the RFNM failed to appear.

The IMP protocol has since been changed considerably and now permits up to n messages* to be outstanding between pairs of IMPs, regardless of the links used, and regardless of which Hosts are communicating. This last point means that there can be some interference among Hosts connected to the same IMP if the Hosts are communicating with the same destination IMP.

The Host/Host protocol has not been changed to take advantage of the possibility of multiple messages and is unable to achieve maximum possible throughput. In figure 1, the time behavior of a multipacket message is shown as it passes through several IMPs from source to destination.



Packet handling by h IMPs for an m-packet message

Figure 1

---

* currently four, this limit being imposed by IMP buffer space.

Figure 1 is naive in several ways. First, it does not show any interfering traffic, nor have any packets gotten out of order or been routed on alternate paths. Second, all packets are assumed to be the same maximum size. Furthermore, the figure does not show the transmission delay to and from the Hosts. Thus, the results of the analysis will be slightly optimistic.

The logical connection between Hosts will be busy only for m packet times out of h+m-1 packet times. The source IMP will be busy for m packet transmission times sending the message to a neighboring IMP. The last bit of the first packet will arrive at the destination IMP after h packet transmission times (not counting propagation delay) and the remaining m-1 packets will complete arrival after m-1 packet transmission times. The source Host will be permitted to transmit another message after it receives a RFNM from the destination IMP. The RFNM is actually sent after the message has been reassembled, the first packet has been delivered, and the destination IMP has sufficient free buffer space for another maximum length multipacket message.* Thus a new transmission cannot occur until h+m-1 packet times, at least, so the fraction of busy time is just m/(h+m-1).

The actual bandwidth between IMPs is reduced from 50 kb** to 40 kb by overhead bits needed for Host/Host, IMP/IMP control. IMPs send periodic routing messages to all their neighbors (every .640 seconds)*** and these consume further bandwidth. We can estimate the nominal fraction of 50 kb/sec bandwidth available from source to destination IMP and multiply this by the fractional busy time per connection to obtain an optimistic bound on maximum throughput per connection.

---

* If after 1 second no space is available, the RFNM is sent anyway.

** Some IMPs have 230 kb/sec lines, or 9.6 kb/sec, but most have 50 kb/sec.

***This interval is a function of line speed and load and may be as low as 128 ms.

Analysis of Expected Throughput Bounds

Let T be the number of bits of text to be transmitted by a Host whose natural word length is W bits. The Host/Host message format includes a 32 bit leader followed by a 40 bit prefix, followed by the text to be sent. We will assume that a sending Host will transmit an integral number of its words, including the 72 bits preceding the text of the message. Furthermore, the Host/IMP interface appends a one bit to each message, followed by as many zeroes as are needed to make the ensemble an integral number of 16 bit IMP words (the IMP is a Honeywell 316 or 516 computer).

The total number of bits in a Host message whose text contains T bits is given by equation 1.

$$M(T,W) = B_1(T) + B_2(T,W) + B_3(T,W) \qquad (1)$$

where $B_1(T) = T + 72$

$$B_2(T,W) = - B_1(T) \bmod W$$

$$B_3(T,W) = 1 + (- B_1(T) - B_2(T,W) - 1) \bmod 16$$

$B_2(T,W) = W - B_1(T) \text{ MOD } W$
$B_3(T,W) = 1 + (15-(B_1(T) + B_2(T,W)) \text{ MOD } 16)$

2

$B_1(T)$ is the number of bits in the Host message including leader, prefix, and text. $B_2(T,W)$ is the number of bits needed to make $B_1(T)$ an integral number of Host words, and $B_3(T,W)$ is the number of bits needed to make the total an integral number of 16 bit IMP words.

The M(T,W) bits are converted to packets in the following way. The 32 bit leader is removed and the remaining words are divided into packets containing no more than 1008 bits of data, each preceded by an 96 bit header which includes the data from the 32 bit leader. When these packets are transmitted to a neighboring IMP, they are enclosed

PACKET SIZE
up to 1104 bits

168 overhead bits The Packet on the line is 1176 bits

in a line control envelope consisting of 48 bits of control octets and a 24 bit cyclic checksum. We can compute the number of bits required to carry all the packets as follows:

$$P(T,W) = \left(\frac{M(T,W)-33}{1008} + 1 \right) \times 168 + M(T,W) - 32 \qquad (2)$$

The line transmission efficiency when transmitting T bits of Host text is given by

$$LTE(T,W) = T/P(T,W) \qquad (3)$$

The expected fraction of time a logical link, which is H hops long, can be busy carrying a Host message of T text bits is given by

$$EBF(T,W,H) = \frac{P(T,W)}{H*\min[P(T,W),\ 1176] + \max\ [P(T,W)-1176\ ,0]} \qquad (4)$$

Note that if data text is 'DLE' then doubled on the line So this calculation assumes 'DLE' never sent as data.

EBF(T,W,H) is a refinement of the fraction computed earlier (m/(m+h-1)). The numerator of EBF(T,W,H) is just the number of bits which must be transmitted from the source IMP. The denominator uses the min and max functions to deal with the case that a message is less than a full single packet in length. In any case, it takes H hops to deliver the first packet, and the remaining bits follow this packet until the entire message has arrived at the destination IMP.

The routing messages require 1024 bits of text and 136 bits of packet header and line control, and are sent by each IMP to all its adjacent neighbors every .640 seconds. The bandwidth required for routing messages is thus (1160)/.640 = 1.8 kb/sec.

Thus the bandwidth which can be expected for Host messages containing T text bits, sent over H hops, is expressed in equation (5) below.

$$B(T,W,H) = EBF(T,W,H) \times LTE(T,W) \times (50-1.8) \text{ kb/sec} \qquad (5)$$

B(T,W,H) ignores a number of complicating factors:

a) delay for sending RFNM and implicit space reservation for multipacket messages to source IMP.

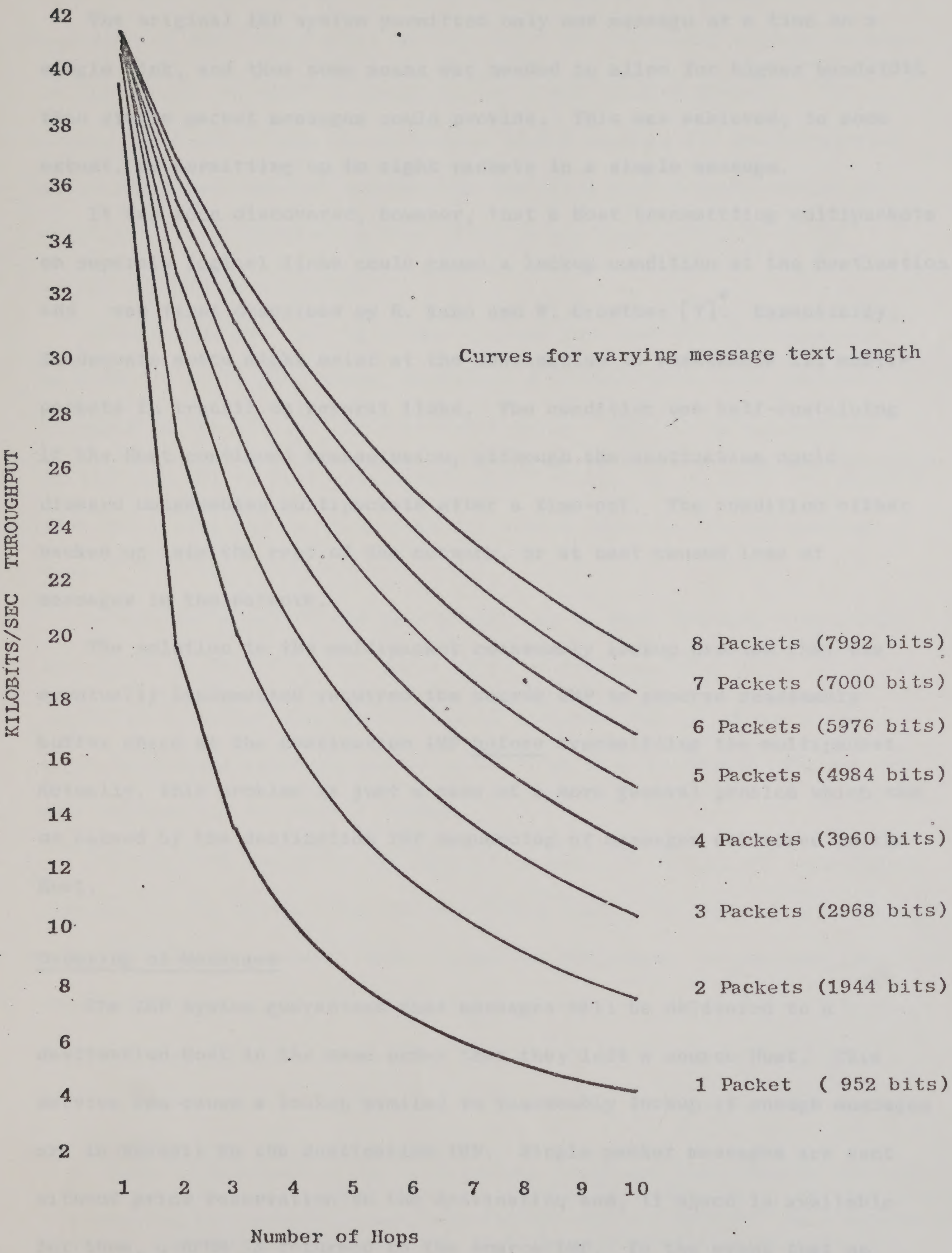
b) propagation delays between Host/IMP and IMP/IMP

c) queueing delays at intermediate IMPs

d) retransmission delays

Nevertheless, B(T,W,H) offers an optimistic estimate of the bandwidth that can be expected using the current ARPANET Host/Host protocol. There is an implicit assumption that packets of a multipacket message are not sent over alternate routes (e.g., two 50 kb/sec paths). Since alternate routing in the IMP subnet is used to avoid congested areas and not to improve bandwidth, this assumption is probably valid for the low traffic densities presently found in the ARPANET.

B(T,W,H) has been plotted in figure 2 for a 32 bit Host (W=32), and a range of message text lengths and Hops. As can be seen, the effect of single message at a time transmission on a single logical connection is very marked for longer and longer hops. The curves would be even lower in the case of a satellite channel owing to the long propagation delay ($\frac{1}{4}$ second up and down) for both the message and the returning RFNM.



Single Link Source to Destination Host Throughput (32 bit word Host)

Figure 2

The Multipacket Message Issue

The original IMP system permitted only one message at a time on a single link, and thus some means was needed to allow for higher bandwidth than single packet messages could provide. This was achieved, to some extent, by permitting up to eight packets in a single message.

It was soon discovered, however, that a Host transmitting multipackets on separate logical links could cause a lockup condition at the destination, and was first described by R. Kahn and W. Crowther [7].* Essentially, inadequate space might exist at the destination to reassemble all multipackets in transit on several links. The condition was self-sustaining if the Host continued transmission, although the destination could discard unassembled multipackets after a time-out. The condition either backed up into the rest of the network, or at best caused loss of messages in the network.

The solution to the multipacket reassembly lockup problem that was eventually implemented required the source IMP to reserve reassembly buffer space at the destination IMP before transmitting the multipacket. Actually, this problem is just a case of a more general problem which can be caused by the destination IMP sequencing of messages delivered to the Host.

Ordering of Messages

The IMP system guarantees that messages will be delivered to a destination Host in the same order that they left a source Host. This service can cause a lockup similar to reassembly lockup if enough messages are in transit to the destination IMP. Single packet messages are sent without prior reservation to the destination and, if space is available for them, a RFNM is returned to the source IMP. In the event that no

---

* Kahn actually knew in 1967 that the condition could occur, but was unable to convince his colleagues until he actually locked up the network by using a message generator to flood the network in March, 1970.

room is available, an implicit reservation request is queued at the destination IMP. When space is available, an allocation message is sent to the source IMP which retransmits the single packet message. The source IMP keeps a copy of the single packet message for retransmission until it either receives a RFNM from the first copy transmitted or an allocate message indicating that there is now room available for a second copy to be accepted.*

This scheme can fail if either a given Host has too many messages in transit, or if many Hosts, served by different IMPs, have too many messages in transit for the same destination. This is so because the destination IMP will accept packets which arrive out of order and buffers them until they can be re-ordered for transmission to the destination Host.

Presently, a source IMP only permits up to four messages (regardless of length) to be in transit for a given destination at a time. This essentially reduces the probability of a lockup, but it is not zero, since sufficient messages may be outstanding from different IMPs for the same destination to cause a lockup.

Such lockups are protected against as well, by timing out undelivered messages at the destination and discarding them. The timeout is on the order of tens of seconds. Even though the IMP subnet can recover from such conditions, it is apparent that Hosts must be prepared to retransmit messages occasionally to recover from message loss caused by deliberate discarding of messages at the destination or by catastrophic failures in which an IMP loses all its packets upon crashing.

---

* R. Kahn, L. Kleinrock, and H. Opderbeck point out that IMPs do not accept out-of-order packets, but do send allocates back for them. If room is also allocated for unreceived but anticipated in-order packets, no lockup will occur. If this step is omitted, then the implementation may fail.

Host Retransmission, Sequencing, and Duplicate Detection

The Host/Host protocol does not provide for retransmission. If it did, however, then this would require that the destination Host detect duplicate transmissions and also verify sequencing of arriving messages since the destination IMP cannot, in the current scheme, detect that a Host has sent a duplicate message.

If this line of reasoning is pursued, it becomes evident that sequencing of messages by the destination IMP is redundant and could be eliminated. Furthermore, with the elimination of ordering, multipacket messages could also be eliminated so long as Hosts were permitted to transmit a sufficient number of single packet messages to achieve maximum potential bandwidth.

Along with Host retransmission, it is necessary to introduce some kind of end-to-end positive acknowledgment. The RFNM is currently sent by the destination IMP to the source Host and is taken to mean that a message has been successfully delivered to the destination Host (for multipacket messages, the RFNM is sent after the first packet has been delivered). It seems sensible to arrange a Host level acknowledgment which confirms delivery. In this case, the RFNM could also be eliminated. One might use RFNM's optionally as a debugging tool, to be turned off and on at will.

Statistics taken from the ARPANET indicate that Host retransmission would rarely be required on account of message loss, but this is partly because of the retransmission and reservation facilities in the current IMP system.

Host Retransmission, Sequencing, and Duplicate Detection

The host/host protocol does not provide for retransmission. If it did, however, then this would require that the destination host detect duplicate transmissions and also verify sequencing of arriving messages since the destination IMP cannot, in the current scheme, detect that a host has sent a duplicate message.

If this kind of mechanism is pursued, it becomes evident that sequencing of messages by the destination IMP is redundant and could be eliminated. Furthermore, with the elimination of ordering, multipacket messages could also be eliminated so long as hosts were permitted to transmit a sufficient number of single packet messages to achieve adequate bandwidth.

Along with host retransmission, it is necessary to introduce some kind of end-to-end positive acknowledgment. The RFNM is currently sent by the destination IMP to the source host and is taken to mean that a message has been successfully delivered to the destination host (for multipacket messages, the RFNM is sent after the first packet has been delivered). It seems sensible to arrange a host level acknowledgment which confirms delivery. In this case, the RFNM could also be eliminated, but might be kept optionally as a debugging tool, to be turned off and on at will.

Statistics taken from the ARPANET indicate that host retransmission would rarely be required on account of message loss, but this is mainly because of the retransmission and reservation facilities in the current IMP system.

Flow Control

If all end-to-end retransmission, duplicate detection, and sequencing are performed by Hosts, then it is essential that the source and destination Hosts agree upon a maximum number of packets (or bits, octets, etc.) that can be outstanding at one time. Otherwise, the destination Host may experience lockup problems similar to those found now in the destination IMP.

The current Host/Host flow control scheme has several weaknesses. First, it requires that special control messages be sent on a control connection which is distinct from the connection on which data is transmitted. Second, it is an incremental scheme in which the destination Host allocates a certain number of bits and messages which may be sent by the source. Both source and destination Hosts decrement these counts as messages are sent and received. To maintain throughput, the destination must periodically send allocations to the source to replenish its available buffer space. Destinations with small amount of buffer space (e.g., Terminal IMPs or TIPs) must do this fairly frequently and thus generate considerable control traffic. Third, the loss of an allocation or the duplication of one can cause loss of synchrony between source and destination.

In an earlier paper [9], the author and R. Kahn propose a more robust flow control scheme including ideas found in the French CYCLADES network [10]. Essentially, the receiver allocates a window representing the span of sequence numbers that the sender may transmit. Acknowledgments from the receiver to the sender indicate the largest sequence number received so far (implicitly acknowledging all those preceding), and also indicate the current width of the window. The sender immediately knows which sequence

numbers can be sent next. The scheme also allows for duplicate detection and reordering of messages.

Acknowledgment and flow control information is sent "piggy-back" with data flowing in the reverse direction of a full duplex logical connection so that a separate control connection is not needed for this purpose. For example, a message is sent with sequence number M and length L in octets. The receiver will respond with acknowledgment of sequence number M+L and window size W. The sequence number of each message is the sequence number of the previous message plus its length in octets.

The receiver can vary the size of W without any serious adverse effect, and can survive the receipt of duplicates or the loss of messages due to the retransmission and duplicate detection permitted by the scheme. The sender is not permitted to transmit a message whose sequence number would exceed the sum of the last sequence number acknowledged plus the current window size, W, modulo the maximum sequence number plus one.

Arbitrary Message Lengths

Until now, it has been implicit that multipacket messages are unnecessary for maintaining high throughput, as long as sufficient packets can be sent to fill the delay pipeline from source to destination Host. If the IMP system were programmed with knowledge of the Host/Host protocol so that it could create a properly formatted Host/Host header for each packet it transmits, given the initial header of an arbitrarily long message, then packets could be delivered out of order to the destination Host, so long as each correctly identified the range of sequence numbers contained in each packet. Since each octet of a message has an implicit sequence number, this would not be difficult to compute. An idea similar

to this is found in the Very Distant Host Reliable Transmission Package [appendix F, 5] in the current ARPANET, except in this case, a Host must know about IMP packet format. It is debatable whether this would be a good idea, since changes in Host/Host protocol would require changes in IMP programming, but if it were implemented, then Hosts could send arbitrarily long messages. The destination Host would receive a collection of single packet messages which it would then sequence as if they had been sent that way by the source Host in the first place.

Simplex versus Full-Duplex Logical Connections

The present Host/Host protocol implements simplex connections. The usage over the last five years seems to indicate that most often, two simplex connections are set up to act as a full duplex connection. If Host level acknowledgments and flow control are implemented, then it is natural for them to be carried in the reverse direction of a full duplex logical connection. Furthermore, terminal to Host connections are necessarily full-duplex to allow for data to move in both directions. Finally, by embedding control in the headers of returning traffic on the full duplex connection, the need for a separate control connection could be eliminated.

Connection Set-up

The current Host/Host protocol uses control messages sent on a special control connection to establish new connections. The procedure is called the Initial Connection Protocol or ICP [11]. The protocol is symmetric and requires that information be exchanged by both Hosts as to the names of the sockets at either end of the connection. This exchange precedes any flow of data. Other control messages are exchanged which determine the buffer space available at the receiver.

A proposal by D. Walden [12] suggests that this is largely unnecessary, as long as both sides can simultaneously send data identifying the source and destination sockets (Walden calls them Ports) along with the text of the messages.

A post office analogy is useful to describe what is intended. The source Host writes a letter and encloses it in an envelope addressed to the destination port with a return port address. Either the destination port is willing to receive or it is not (e.g. it may not even be known to the destination Host). In the former case, the letter is acknowledged in the usual fashion. In the latter case, the letter is not acknowledged (port unprepared to receive), or it is rejected ("address unknown").

Since port addresses may be dynamically assigned to processes in a destination Host, it will probably be necessary to include a formal control exchange which indicates to the sender that a receive port is being closed, and the sender would be expected to acknowledge this. Similarly, the sender may end a transmission with the indication that the send port is being closed and the receiver would similarly acknowledge. Since Hosts do the sequencing, there can be no confusion as to when the closure is to take place. The rejection of an initial transmission can be made to look like the closure of the destination port so that the number of distinct control messages can be kept to a minimum. This method is similar to the one currently used in the ARPANET, but could be carried out via control bits in the Host level messages and thus eliminate the need for a special control connection.

Summary

NOT so clear

Arguments have been presented in this paper which show that multi-packet reassembly is not the best vehicle for achieving high throughput from Host to Host. The elimination of IMP reassembly as well as message sequencing by the destination IMP can permit considerable simplification of the IMP protocols, while simultaneously placing the burden of buffering, duplicate detection, and sequencing of messages on the Hosts which have the buffer space for this purpose.

Arbitrarily long messages could be sent by Hosts, at the expense of IMP knowledge of Host protocol. Eliminating the ordering requirement at the destination IMP also eliminates serious potential lockup conditions.

good

Host level positive acknowledgments can eliminate the erroneous use of the RFNM for this purpose, and permit a more robust protocol which need not depend upon perfect performance without message loss by the IMP subnet.

Good

Full duplex logical connections between ports in Hosts are more natural than the simplex connections presently used, and facilitate the elimination of the special control connection required in the current Host protocol.

Unresolved Problems and Issues

Even if a source and destination Host have adequate buffer space to permit a large number of messages (or packets, or octets) to be outstanding between them, the IMP subnet must have a way of combatting congestion which may result from too rapid influx of data from a source Host, or from momentary congestion owing to the confluence of excessive traffic heading in the same direction, possibly to the same destination. Alternate

routing strategies can help, but not completely solve the problem. One possibility is to insist that source Hosts monitor actual throughput achieved over the last few seconds (milliseconds?) and adjust output rate accordingly. Destination Hosts can monitor this throughput as well, and adjust the receive buffer space it allocates to the sender to reduce unnecessary retransmissions. The IMPs can simply discard traffic which cannot be buffered, knowing that the source will retransmit. IMPs which discard packets to eliminate congestion could even send short warning messages to source or destination (or both) to stimulate adjustment. This is a very sticky problem and involves issues such as payment by Hosts for retransmission. Most strategies in use today involve limiting, a priori, the amount of data which a source Host is allowed to send (e.g., isarhythmic network proposed by Davies [13]; maximum of n messages allowed by ARPANET IMPs). Measurement of throughput achieved by source and destination Hosts may be a good strategy in any case since it serves as a measure of quality of service provided by the packet switching network.

In the ARPANET, the TELNET protocol [14] for terminal to Host communication has needed some way of signalling the Host in which the serving process resides that any accumulated data should be discarded up to the point of the "interrupt signal." This facility permits a remote user to abort or recapture control from an uncooperative serving process which has stopped accepting data. The current scheme involves the use of a special interrupt signal sent on the control connection, but there is a problem of synchronizing the interrupt request with the data in the pipeline. This signal could be carried in the control field of a Host message and would participate in the sequence numbering of the data, thus routing strategies can help, but not completely solve the problem. One possibility is to insist that source hosts monitor actual throughput achieved over the last few seconds (milliseconds?) and adjust output rate accordingly. Destination hosts can monitor this throughput as well, and adjust the receive buffer space it allocates to the source to reduce unnecessary retransmissions. The IMPs can simply discard traffic which cannot be buffered, knowing that the source will retransmit. (The other several proposals to eliminate congestion could even send choke messages addressed to source or destination (or both) to eliminate retransmission. This is a very sticky problem and involves issues such as payment by hosts for retransmission. Most interesting is one scheme involving tracking, a priori, the amount of data which a source host is allowed to send (e.g., isarithmic networks proposed by Davies [15]) mentioned by a message allowed by ARPANET IMPs). Measurement of throughput achieved by source and destination hosts may be a good strategy in any case since it serves as a measure of quality of service provided by the packet switching network.

In the ARPANET, the TELNET protocol [16] for terminal to host communication has needed some way of signalling the host to which the service process applies that any accumulated data should be discarded up to the point of the "interrupt signal." This facility permits a remote user to abort an excessive output from an uncooperative service process which has stopped accepting data. The current scheme involves the use of a special interrupt signal sent on the control connection, but there is a problem of synchronizing the interrupt request with the data in the pipeline. This signal could be carried in the control field of a host message and could participate in the sequence numbering of the data, thus providing for synchronization. Since the Host operating system would process the message header before passing the data to the receiving port, the interrupt could bypass processing by the receiving process and thus provide the desired interrupt-like effect.

There are undoubtedly many other problems and issues which could not be mentioned in the scope of this paper, and the author would be pleased if these and the preceding commentary will stimulate discussion and thus further the general understanding of protocol requirements for distributed computer networks.

Acknowledgments:

Throughput and delay analysis: some of the basic ideas in this analysis are due to J. McQuillan (Bolt, Beranek, and Newman). Single packet re-ordering lockup: first called to the author's attention by P. Higginson (University of London). Host-Host Protocol Design: developed largely under the auspices of the International Network Working Group (IFIP TC6.1), and the author especially acknowledges contributions by R. Kahn, R. Metcalfe, L. Pouzin and H. Zimmerman, as well as S. Crocker, A. McKenzie, and R. Scantlebury. Numerous omissions and misstatements were detected by R. Kahn, L. Kleinrock and H. Opderbeck.

The author is grateful for the support of the Defense Advanced Research Projects Agency under contract DAHC-15-73C-0435.

References

1. McKenzie, A. "HOST/HOST Protocol for the ARPA Network," Current Network Protocols, Network Information Center, Stanford Research Institute, Menlo Park, California, January 1972 (NIC8246).
2. Carr, S. and S. Crocker, V. Cerf, "HOST-HOST Communication Protocol in the ARPA Network," AFIPS 1970, SJCC Proceedings, Vol. 36, Atlantic City, AFIPS Press, New Jersey, pp. 589-597 [somewhat out of date].
3. Crocker, S. D., and J. Heafner, R. Metcalfe, J. Postel, "Function-Oriented Protocols for the ARPA Computer Network," AFIPS 1972 SJCC Proceedings, Vol. 40, Atlantic City, AFIPS Press, New Jersey, pp. 271-279.
4. Heart, F. E., and R. E. Kahn, et al, "The Interface Message Processor for the ARPA Computer Network," AFIPS 1970 SJCC Proceedings, Vol. 36, Atlantic City, AFIPS Press, New Jersey, pp. 551-567.
5. Bolt, Beranek and Newman, Inc.,"Specification for the Interconnection of a Host and an IMP," BBN Report 1822, April 1973 (Revised).
6. Roberts, L. and B. Wessler, "Computer Network Development to Achieve Resource Sharing," AFIPS 1970, SJCC Proceedings, Vol. 36, Atlantic City, AFIPS Press, New Jersey, pp. 543-549.
7. Kahn, R. E. and W. Crowther, "Flow Control in a Resource Sharing Computer Network," Second Symposium on Problems in the Optimization of Data Communications Systems, Palo Alto, October 1971, p. 108-116 [also IEEE Transactions on Communication, June 1972].
8. Kahn, R. E., "Resource Sharing Communication Networks," IEEE Proceedings, Nov. 1972.
9. Cerf, V. G. and R. E. Kahn, "A Protocol for Packet Network Intercommunication," IEEE Transactions on Communication, to appear May 1974.

10. Pouzin, L., "Presentation and Major Design Aspects of the CYCLADES Computer Network," Data Networks: Analysis and Design, Third Data Communications Symposium, St. Petersburg, Florida, November 1973, pp. 80-87.

11. Postel, J., "Official Initial Connection Protocol," Current Network Protocols, Network Information Center, Stanford Research Institute, Menlo Park, California, January 1972 (NIC 7101).

12. Walden, D., "A System for Interprocess Communication in a Resource Sharing Computer Network," Comm. of the ACM, 15, 4, April 1972, pp. 221-230 (NIC 9926).

13. Davies, D., "Communication Networks to Serve Rapid Response Computers," Information Processing 68, Proceedings of IFIP Congress 1968, Vol. 2, Edinburgh, Scotland, 1968, North-Holland Publishing Co., Amsterdam, 1969, p. 650-658.

14. McKenzie, A. "TELNET Protocol Specification," Current Network Protocols, Network Information Center, Stanford Research Institute, Menlo Park, California, August 1973 (NIC 18639, NIC 18638 - Revisions).